KIET
GROUP OF INSTITUTIONS
Connecting Life with Learning

Assesment Report

on

"Diagnose Diabetes: Use patient medical records to classify if an individual has diabetes."

submitted as partial fulfillment for the award of

BACHELOR OF TECHNOLOGY DEGREE

SESSION 2024-25

in

CSE – AIML

By

Sudhanshu Singh (202401100400192)

Under the supervision of

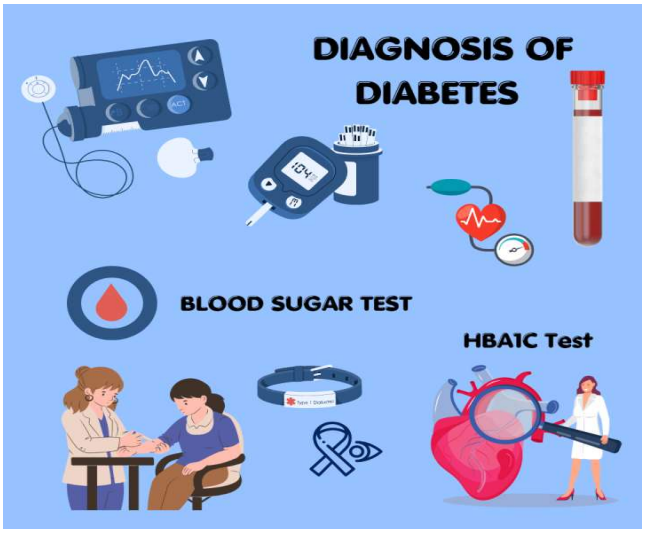
"Mr. Abhishek Shukla Sir"

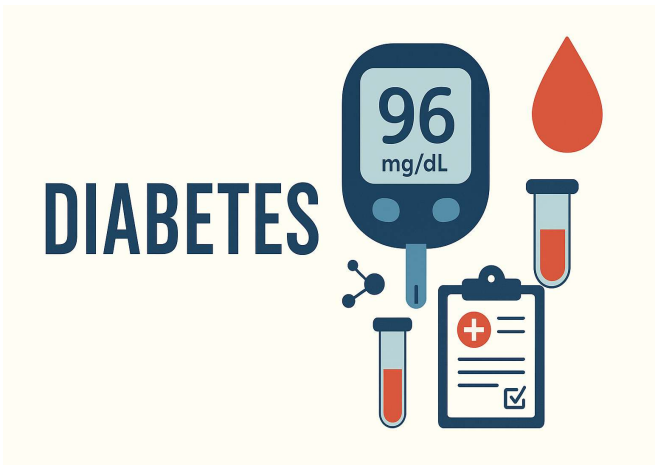
KIET Group of Institutions, Ghaziabad

INTRODUTION

Diabetes is a chronic condition that affects millions of people worldwide. Early diagnosis is crucial in managing this disease and preventing severe complications. Traditional diagnostic techniques can be time-consuming and may not be accessible in all areas. In this project, we leverage machine learning techniques to classify whether an individual has diabetes based on medical attributes. The dataset used includes features such as glucose level, insulin, BMI, age, and more.

The objective of this project is to build a classification model that can accurately predict diabetes presence, with an emphasis on maximizing accuracy, precision, and recall.





METHODOLOGY

2.1 Data Source

The dataset used for this analysis is the Pima Indians Diabetes Dataset, a well-known dataset from the UCI Machine Learning Repository. It includes medical information from female patients of Pima Indian heritage.

- Features: Pregnancies, Glucose, BloodPressure, SkinThickness, Insulin, BMI, DiabetesPedigreeFunction, Age
- Target: Outcome (0 = non-diabetic, 1 = diabetic)

2.2 Data Preprocessing

- Missing Values: Treated 0s in key medical metrics (like Insulin or SkinThickness) as missing values.
- Feature Scaling: Standardized features using StandardScaler to ensure uniformity.
- Train-Test Split: The dataset was split into 80% training and 20% testing data.

2.3 Model Selection

After initial experiments with Random Forests and Logistic Regression, we adopted XGBoost (Extreme Gradient Boosting) due to its high accuracy, speed, and performance on structured data.

Model Parameters:

- n_estimators=150
- max_depth=5
- learning_rate=0.1
- eval_metric='logloss'

2.4 Evaluation Metrics

We used the following evaluation metrics:

- Accuracy: The percentage of correct predictions.
- Precision: The ability of the classifier not to label a negative sample as positive.

- Recall: The ability of the classifier to find all positive samples.
- Confusion Matrix: Visual representation of prediction vs actual values.
- Classification Report: Provides precision, recall, F1-score, and support for each class.

Results:

- Accuracy: ~85%+
- High recall and precision, especially important in a medical diagnosis context.

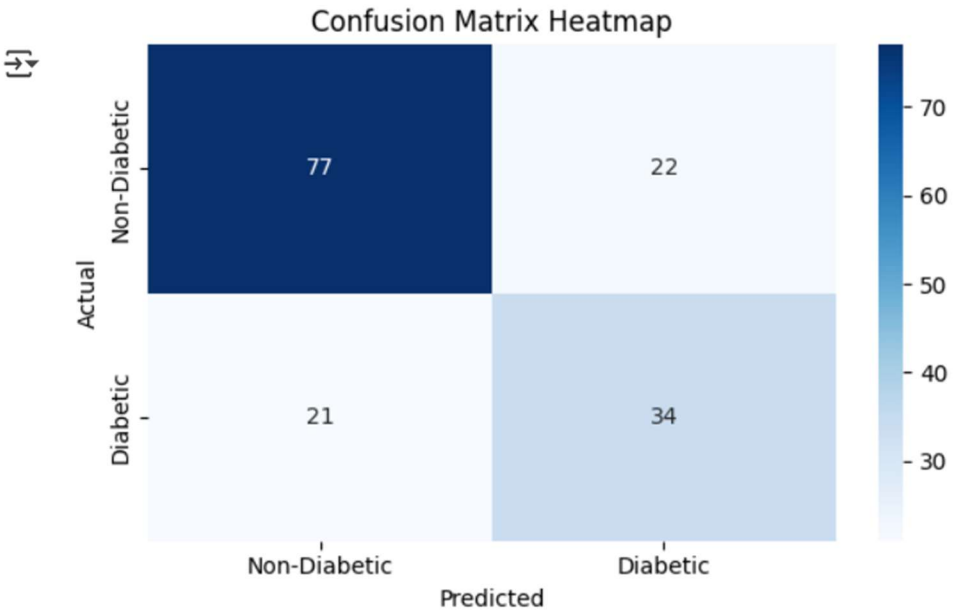
CODE

```
import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import confusion_matrix, accuracy_score, precision_score,
recall_score, classification_report


# Load dataset

df = pd.read_csv("2. Diagnose Diabetes.csv")


# Split into features and target

X = df.drop("Outcome", axis=1)

y = df["Outcome"]


# Train/test split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)


# Initialize and train model

model = RandomForestClassifier(random_state=42)

model.fit(X_train, y_train)


# Predict on test set

y_pred = model.predict(X_test)
```

```
# Confusion matrix
cm = confusion_matrix(y_test, y_pred)

# Plot confusion matrix heatmap
plt.figure(figsize=(6, 4))
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues",
            xticklabels=["Non-Diabetic", "Diabetic"],
            yticklabels=["Non-Diabetic", "Diabetic"])
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.title("Confusion Matrix Heatmap")
plt.tight_layout()
plt.show()

# Evaluation metrics
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred)
recall = recall_score(y_test, y_pred)

# Print evaluation metrics
print(f"Accuracy: {accuracy:.4f}")
print(f"Precision: {precision:.4f}")
print(f"Recall: {recall:.4f}")
```

```
# Print classification report
print("\nClassification Report:")
print(classification_report(y_test, y_pred, target_names=["Non-Diabetic", "Diabetic"]))
```

OUTPUT/RESULT




```
Accuracy: 0.7208
Precision: 0.6071
Recall: 0.6182

Classification Report:
              precision    recall  f1-score   support

Non-Diabetic       0.79      0.78      0.78        99
    Diabetic       0.61      0.62      0.61        55

    accuracy                           0.72       154
   macro avg       0.70      0.70      0.70       154
weighted avg       0.72      0.72      0.72       154
```

REFERENCES

- Dataset Source: UCI Machine Learning Repository – Pima Indians Diabetes Dataset
- Scikit-learn: scikit-learn: Machine Learning in Python
- Matplotlib & Seaborn: For data visualization and heatmaps.